LUX ET VERITAS

Case-Based Reasoning: A Research Paradigm

Stephen Slade

YALEU/CSD/RR #644

August 1988

YALE UNIVERSITY
DEPARTMENT OF COMPUTER SCIENCE

4

Case-Based Reasoning: A Research Paradigm

Stephen Slade

YALEU/CSD/RR #644

August 1988

This work was supported in part by the Advanced Research Projects Agency of the Department of Defense and the Office of Naval Research under contract N00014-85-K-0108, and by the Air Force Office of Scientific Research under contract AFOSR-87-0295.

ADA202363

SECURITY CLASSIFICATION OF THIS PAGE *(When Data Entered)*

REPORT DOCUMENTATION PAGE	READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER #644 | 2. GOVT ACCESSION NO	3. RECIPIENT'S CATALOG NUMBER
4. TITLE *(and Subtitle)* Case-Based Reasoning: A Research Paradigm	5. TYPE OF REPORT & PERIOD COVERED Research Report 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Stephen Slade	8. CONTRACT OR GRANT NUMBER(s) N00014-85-K-0108
9. PERFORMING ORGANIZATION NAME AND ADDRESS Yale University Computer Science Department 51 Prospect Street New Haven, CT 06520	10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Advanced Research Projects Agency 1400 Wilson Boulevard Arlington, VA 22209	12. REPORT DATE August 88 13. NUMBER OF PAGES 38
14. MONITORING AGENCY NAME & ADDRESS*(if different from Controlling Office)* Office of Naval Research Information Systems Program Arlington, VA 22217	15. SECURITY CLASS. *(of this report)* unclassified 15a. DECLASSIFICATION DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT *(of this Report)*

Approved for public release: Distribution unlimited

17. DISTRIBUTION STATEMENT *(of the abstract entered in Block 20, if different from Report)*

18. SUPPLEMENTARY NOTES

19. KEY WORDS *(Continue on reverse side if necessary and identify by block number)*
case-based reasoning
knowledge representation
expert systems
cognitive modelling
memory organization

20. ABSTRACT *(Continue on reverse side if necessary and identify by block number)*

Expertise comprises experience. In solving a new problem, we rely on past episodes. We need to remember what plans succeed and what plans fail. We need to know how to modify an old plan to fit a new situation. Case-based reasoning is a general paradigm for reasoning from experience. Case-based reasoning assumes a memory model for representing, indexing and organizing past cases, and a process model for retrieving and modifying old cases, and assimilating new ones.

Case-based reasoning provides a scientific cognitive model. The research issues for case-based reasoning include representation of episodic knowledge, memory organization, indexing, planning, case modification, and learning. In addition, computer implementations of case-based reasoning address many of the technological short-comings of standard rule-based expert systems. These engineering concerns include knowledge acquisition and robustness.

In this report we review the history of case-based reasoning including research conducted at the Yale AI Project and elsewhere.

OFFICIAL DISTRIBUTION LIST

Defense Documentation Center 12 copies
Cameron Station
Alexandria, Virginia 22314

Office of Naval Research 2 copies
Information Systems Program
Code 437
Arlington, Virginia 22217

Dr. Judith Daly 3 copies
Advanced Research Projects Agency
1400 Wilson Boulevard
Arlington, Virginia 22209

Office of Naval Research 1 copy
Branch Office - Boston
495 Summer Street
Boston, Massachusetts 02210

Office of Naval Research 1 copy
Branch Office - Pasadena
1030 East Green Street
Pasadena, California 91106

Mr. Steven Wong 1 copy
New York Area Office
715 Broadway - 5th Floor
New York, New York 10003

Naval Research Laboratory 6 copies
Technical Information Division
Code 2627
Washington, D.C. 20375

Dr. A.L. Slafkosky 1 copy
Commandant of the Marine Corps
Code RD-1
Washington, D.C. 20380

Office of Naval Research 1 copy
Code 455
Arlington, Virginia 22217

Office of Naval Research 1 copy
Code 458
Arlington, Virginia 22217

Naval Electronics Laboratory Center 1 copy
Advanced Software Technology Division
Code 5200
San Diego, California 92152

Dr. Sidney Berkowitz 1 copy
Naval Ship Research and Development
Computation and Mathematics Department
Bethesda, Maryland 20084

Dr. A. Sears 2 copies
ISTO-DARPA
1400 Wilson Boulevard
Arlington, Virginia 22209

Professor John Kender 1 copy
Computer Science Department
Columbia University
New York, New York 10027

Office of Naval Research 1 copy
Assistant Chief for Technology
Code 200
Arlington, Virginia 22217

Computer Systems Management, Inc. 5 copies
1300 Wilson Boulevard, Suite 102
Arlington, Virginia 22209

Ms. Robin Dillard 1 copy
Naval Ocean Systems Center
Artificial Intelligence Branch (Code 444)
271 Catalina Boulevard
San Diego, California 92152

Dr. William Woods 1 copy
ON Technology, Inc.
One Cambridge Center
Cambridge, MA 02142

Professor A. Van Dam 1 copy
Dept. of Computer Science
Brown University
Providence, RI 02912

Professor Eugene Charniak 1 copy
Dept. of Computer Science
Brown University
Providence, RI 02912

Professor Robert Wilensky 1 copy
Univ. of California
Elec. Engr. and Computer Science
Berkeley, CA 94707

Professor Allen Newell 1 copy
Dept. of Computer Science
Carnegie-Mellon University
Schenley Park
Pittsburgh, PA 15213

Professor David Waltz 1 copy
Thinking Machines, Inc.
245 First Street
Cambridge, MA 02142

Professor Patrick Winston 1 copy
MIT
545 Technology Square
Cambridge, MA 02139

Professor Marvin Minsky 1 copy
MIT
545 Technology Square
Cambridge, MA 02139

Professor Negroponte 1 copy
MIT
545 Technology Square
Cambridge, MA 02139

Professor Jerome Feldman 1 copy
Dept. of Computer Science
University of California
Berkeley, CA 94720

Dr. Nils Nilsson 1 copy
Department of Computer Science
Stanford University
Stanford, CA 94025

Dr. Alan Meyrowitz 1 copy
Office of Naval Research
Code 437
800 N. Quincy Street
Arlington, VA 22217

Lt. Col. Robert Simpson 1 copy
ISTO-DARPA
1400 Wilson Blvd
Arlington, VA 22209

Dr. Edward Shortliffe 1 copy
Stanford University
MYCIN Project TC-117
Stanford Univ. Medical Center
Stanford, CA 94305

Dr. Douglas Lenat 1 copy
MCC
9430 Research Boulevard
Echelon Building #1, Suite 200
Austin, TX 78759

Dr. M.C. Harrison 1 copy
Courant Institute Mathematical Science
New York University
New York, NY 10012

Dr. Morgan 1 copy
University of Pennsylvania
Dept. of Computer Science & Info. Sci.
Philadelphia, PA 19104

Mr. Fred M. Griffee 1 copy
Technical Advisor C3 Division
Marine Corps Development
and Education Command
Quantico, VA 22134

Dr. Abraham Waksman 1 copy
Program Manager
AFOSR/NM
Bolling Air Force Base
Building 410
Washington, DC 20332

Capt. Jim Parlett 1 copy
AFHRL/IDI
Brooks Air Force Base, TX 78235

Case-based Reasoning: A Research Paradigm

Stephen Slade
Yale Artificial Intelligence Project
Yale Department of Computer Science
Box 2158
New Haven, Conn. 06520

YALEU/CSD/RR #644
August 1988

Abstract

Expertise comprises experience. In solving a new problem, we rely on past episodes. We need to remember what plans succeed and what plans fail. We need to know how to modify an old plan to fit a new situation. *Case-based reasoning* is a general paradigm for reasoning from experience. Case-based reasoning assumes a memory model for representing, indexing and organizing past cases, and a process model for retrieving and modifying old cases, and assimilating new ones.

Case-based reasoning provides a scientific cognitive model. The research issues for case-based reasoning include representation of episodic knowledge, memory organization, indexing, planning, case modification, and learning. In addition, computer implementations of case-based reasoning address many of the technological short-comings of standard rule-based expert systems. These engineering concerns include knowledge acquisition and robustness.

In this report we review the history ~~of~~ case-based reasoning including research conducted at the Yale ~~AI~~ Project and elsewhere.

[illegible] Abst[illegible]) [illegible])

human cognitive [illegible] IPP [illegible]

IPP (Integrated Partial Parsing), CH[illegible] [illegible] program (KR)

[illegible]

[*] This work was supported in part by the Advanced Research Projects Agency of the Department of Defense and the Office of Naval Research under contract N00014-85-K-0108, and by the Air Force Office of Scientific Research under contract AFOSR-87-0295.

Contents

1 **Introduction** 1

2 **Models of Memory** 1

2.1 Semantic and Episodic Memory 2

2.2 Conceptual Memory 3

2.3 Scripts, MOPs, and Reminding 4

2.4 Process Model 6

2.5 Psychological Issues 9

3 **Computer Models** 10

3.1 Expert Systems: Rules vs. Cases 11

3.2 Case-based Systems 14

4 **Summary** 18

5 **References** 20

A **Appendix: Case-based learning in IPP** 27

B **Appendix: Case-based planning in CHEF** 34

I have but one lamp by which my feet are guided,
and that is the lamp of experience.
I know no way of judging of the future but by the past.
◇ PATRICK HENRY, *Speech in Virginia Convention, Richmond,*
March 23, 1775

1 Introduction

There are two broad research agendas in artificial intelligence (AI). The first is scientific. We seek to understand the nature of intelligence and human thought. We examine a range of human cognitive behavior, including memory, learning, planning, and problem solving, and look for principles that play general descriptive and explanatory roles. AI shares these scientific ambitions with other cognitive science disciplines.

The second agenda for AI research is technological. We seek to create intelligent artifacts - machines that can perform useful tasks. We wish to develop the technology of intelligence. We want to be able to design and build computer programs that can solve problems and adapt to new situations.

In this report, we discuss *case-based reasoning*, an AI paradigm that addresses both research agendas. Case-based reasoning is a psychological theory of human cognition. It addresses issues in memory, learning, planning, and problem solving. Case-based reasoning also provides a foundation for a new technology of intelligent computer systems that can solve problems and adapt to new situations.

We shall first review the underlying psychological model of case-based reasoning. We then shall examine several computer models that embody the principles of case-based reasoning, contrasting the case-based approach with the rule-based expert system paradigm.

2 Models of Memory

An intelligent being requires knowledge about the world. Knowledge allows a person to plan and solve problems. Knowledge is a resource, a commodity. Memory is the repository of knowledge. The question for the psychologist has been what theory of memory accounts for observed cognitive behaviors. The question for the AI researcher has been how to represent knowledge in a computer program.

2.1 Semantic and Episodic Memory

These questions have converged as AI researchers have attempted to create computer programs that model cognitive processes. A leading theory has been the *semantic network* memory model. Psychologists have devoted much attention to this theory [CQ69, RLN72, Kin72], as have AI researchers [Qui68, Woo75].

Semantic networks typically represent static facts about the world, such as, "Fido is a dog," "A dog is a mammal," and "Mammals have hair." In general, this type of knowledge does not change over time.

Psychologists and AI researchers realized that semantic networks did not account for all the data. First, not all knowledge is in small, static chunks. Memories are variable in size and malleable in content. Second, the semantic network theory did not explain how knowledge is incorporated into memory. Where did the information come from? It is clear that we are not born with an innate knowledge of the world.

To address these and other questions, Tulving proposed a theory of *episodic memory* [Tul72, Tul83] as an adjunct to semantic memory. Tulving described semantic and episodic memory as two complementary information processing systems, both of which perform the following actions:

- receive information from perceptual and cognitive systems.
- process portions of the information.
- communicate information to other behavioral and cognitive systems.

Semantic and episodic memory, according to Tulving, differ by the following features:

- the type of information stored.
- autobiographical reference versus cognitive reference.
- retrieval conditions and consequences.
- volatilitiy of stored information.
- interdependence.

More specifically,

> Episodic memory receives and stores information about temporally dated episodes or events, and temporal-spatial relations among these events. A perceptual event can be stored in the episodic system solely in terms of its perceptible properties or attributes, and it is always stored in terms of its autobiographical reference to the already existing contents of the episodic memory store. The act of retrieval of information from the episodic store, in addition to making the retrieval contents accessible to inspection, also serves as a special type of input into episodic memory and thus changes the contents of the episodic memory store. [Tul72]

Episodic memory provided an account of representing and recalling larger chunks of temporally related information – events, scenes, occurrences, stories. By contrast, "semantic memory is the memory necessary for the use of language. It is mental thesaurus." [Tul72]

2.2 Conceptual Memory

In parallel with the identification of episodic memory by psychologists, AI researchers had arrived at a similar theory for language understanding tasks. Schank and his students had developed natural language systems for representing concepts and understanding single sentences [Sch72, Sch75a]. A sentence such as "John ate a hamburger." could be processed, paraphased, and translated to another language. The next step was to process connected text – paragraphs and stories. For this task, Schank proposed a *conceptual memory* [Sch75b] that combined semantic memory with Tulving's episodic memory.

> The distinction between semantic memory and episodic memory is a false one. We shall argue that what must be present is a lexical memory which contains all of the information about words, idioms, common expressions etc., and which links these to nodes in a conceptual memory, which is language free. We believe that it is semantic memory rather than episodic memory which is the misleading notion. Once we change semantic memory by separating out lexical memory, we are left with a set of associations and other relations between concepts that could only have been acquired by personal experience. We claim that conceptual memory, therefore, is episodic in nature. [Sch75b]

A key feature of Schank's conceptual memory is the notion that information is derived from experience. Knowledge is not innate. A theory of memory must account for the acquisition of knowledge.

2.3 Scripts, MOPs, and Reminding

At this time, Schank and Abelson proposed knowledge structures for representing episodic information [SA75, SA77]. The primary knowledge structure was the *script*. Scripts accounted for information about stereotypical events, such as going to a restaurant, taking a bus, or visiting the dentist. In such common situations, a person has a set of expectations concerning the default setting, goals, props, and behaviors of the other people involved. Scripts are analogous to Minsky's *frames* [Min75] which were proposed in the context of visual processing. It is important to note that scripts are directly related to autobiographical events. Scripts are inherently episodic in origin and use. That is, scripts arise from experience and are applied to understand new events.

Scripts were proposed as a knowledge structure for a conceptual memory. The acquisition of scripts was a result of repeated exposure to a given situation. For example, children learn the restaurant script by going to restaurants over and over again.

As a psychological theory of memory, scripts suggested that people would remember events in terms of their associated script. However, an experiment by Bower, Black, and Turner [BBT79] showed that subjects often confused events that had similar scripts. For example, a subject might mix up scenes from a visit to a doctor's office with a visit to a dentist's office.

These data required a revision in script theory. What knowledge structures would allow for such confusion? What was the underlying process of remembering?

Schank postulated a more general knowledge structure to account for the diverse and heterogeneous nature of episodic knowledge. The new structure was the *Memory Organization Packet* or MOP [Sch79, Sch80, Sch82]. MOPs might be viewed as Meta-scripts. For example, instead of a dentist script or a doctor script, there might be a Professional-Office-Visit MOP that can be instantiated and specified for both the doctor and the dentist.

More important than the MOP knowledge structure was the new emphasis on the basic memory processes of reminding and learning. The early work of Bartlett [Bar32] on remembering had influenced the original design of scripts for story comprehension [Sch75b]. The new focus on reminding raised additional questions about how memory was organized and indexed. Schank illustrated this phenomenon with sample remindings, such as the following, gathered informally from colleagues and students.

> *The steak and the haircut.*
>
> X described how his wife would never make his steak as rare as he liked it. When this was told to Y, it reminded Y of a time, 30 years earlier, when he tried to get his hair cut in a short style in England, and the barber would not cut it as short as he wanted it.
>
> *The sand dollars and the drunk.*
>
> X's daughter was diving for sand dollars. X pointed out where there were a great many sand dollars, but X's daughter continued to dive where she was. X asked why. She said that the water was shallower where she was diving. This reminded X of the joke about the drunk who was searching for his ring under the lamppost because the light was better there even though he had lost the ring elsewhere. [Sch82]

The remindings in these two stories have very little to do with the surface features of the episodes. Cooking a steak and cutting hair are hardly similar events, yet the stories *are* related. These examples illustrate the fact that memory has a complex structure and indexing that allows people to relate a new episode to prior cases through thematic and abstract categories.

Given this richly indexed structure, Schank proposed a theory of learning based on reminding. If we assume that new situations or experiences will remind us of previous cases and events, we can classify a new episode in terms of past cases. The knowledge of the past case, like a script, can guide our behavior. We can rely on the past episode to help us understand a new situation. For example, the second time we ride on an airplane, we will be reminded of our first airplane trip. We can use that experience to remind us to get a boarding pass, find our seat, stow our luggage, fasten the seatbelt, etc.

However, when the new situation does not conform to the prior case, we have a failure. That is, we had an expectation based on a prior event that did not occur in the new situation. Thus, we must classify this new situation as being different from the previous episode. We must remember this new experience. We must learn. Schank termed this process *failure-driven learning* [Sch81]. Returning to our airplane example, if we have flown several times, but then take the air shuttle for the first time, we will have some surprises. Our expectations to have an assigned seat and to buy our ticket ahead of time fail. We must recognize these discrepancies and account for them. We must modify our knowledge structure for airplane rides so that the next time we take the air shuttle, we will know better what to expect.

When we observe a discrepancy between our predictions and some event, we then have something to learn. We need to revise our knowledge structure. The mechanism for updating our knowledge often requires *explanation.* Schank noted that explanation plays a central role in learning and intelligence [Sch86a]. He proposed an explicit knowledge structure, *explanation patterns (XPs)*, that are used to generate, index, and test explanations in conjunction with an episodic memory.

2.4 Process Model

We began with the intent of representing knowledge and thence deriving a theory of memory to account for episodic information. Scripts and MOPs were postulated as knowledge structures for representing experience. However, the knowledge structures provide only part of the answer. We must also specify the processes involved in acquiring and accessing these structures. We require a process model.

In figure 1 (after [RB87]), we present a flow chart that illustrates the basic process of case-based reasoning and learning.

In that figure, boxes represent processes and ovals represent knowledge structures. The process of interpreting and assimilating a new event is broken down in the following steps, starting with an input event at the top of the flow chart.

1. **Assign Indices**. Features of the new event are assigned as indices characterizing the event. For example, our first air shuttle flight might be characterized as "airplane flight."

2. **Retrieve**. The indices are used to retrieve a similar past case from memory. The past case contains the prior solution. In our example, we might be reminded of our last airplane trip.

3. **Modify**. The old solution is modified to conform to the new situation, resulting in a proposed solution. For our airplane case, we would make appropriate modifications to account for changes in various features such as destination, price, purpose of the trip, departure and arrival times, weather, etc.

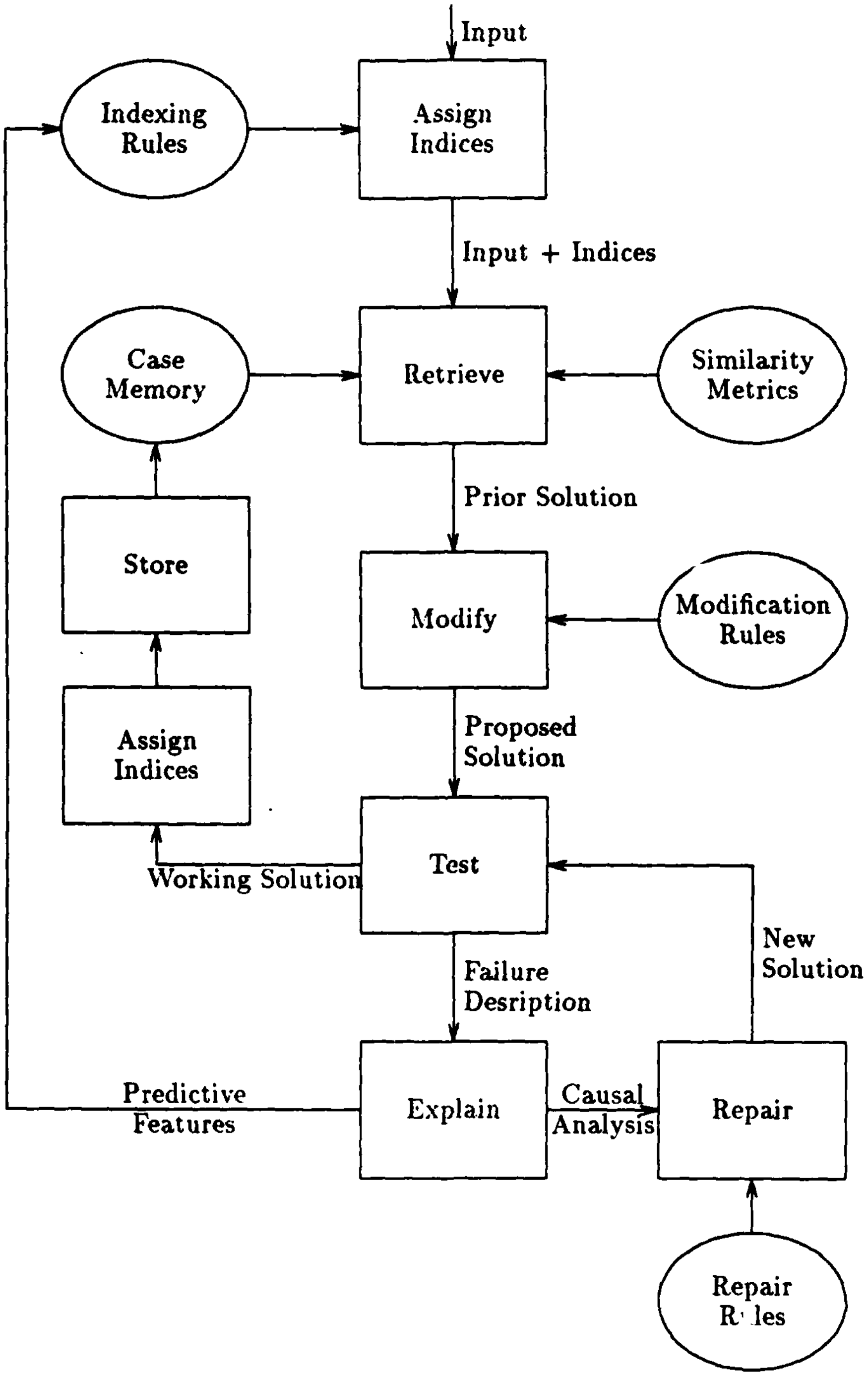

Figure 1: Case-based Reasoning Flow Chart

4. **Test.** The proposed solution is tried out. It either succeeds or fails. Our airplane reminding generates certain expectations, not all of which may be met.

5. If success, then **Assign Indices** and **Store** working solution. The successful plan is then incorporated into the case memory. For a normal airplane trip, there will be few expectation failures and therefore little to make this new trip memorable. It will be just one more instance of the airplane script.

6. If failure, then **Explain** the failure, **Repair** working solution, and **Test again.** The explanation process identifies the source of the problem. The predictive features of the problem are incorporated into the indexing rules to anticipate this problem in the future. The failed plan is repaired to fix the problem and the revised solution is then tested. For our air shuttle example, we realize that certain expectations fail. We learn that we do not get an assigned seat and that we do not have to pay ahead of time. We may decide that taking the air shuttle is more like riding on a train. We then can create a new case in memory to handle this new situation and identify predictive features so that we will be reminded of this episode the next time we take the shuttle.

In support of this process are the following types of knowledge structures, represented by ovals in the figure.

- **Indexing Rules.** These rules identify the predictive features in the input that provide appropriate indices into the case memory. Determining the significant input features is a persistent problem [SCH86b].

- **Case Memory.** This is the episodic memory, which comprises the database of experience.

- **Similarity Metrics.** If more than one case is retrieved from episodic memory, the similarity metrics can be used to decide which case is more like the current situation. For example, in the air shuttle case, we might be reminded of both airplane rides and train rides. The similarity rules might suggest initially to rely on the airplane case.

- **Modification Rules.** No old case is going to be an exact match for a new situation. The old case must be modified to fit. We require knowledge about what kinds of things can be changed and how to change them. For the airplane ride, it is acceptable to ride in a different seat, but it is usually not advisable to change roles from passenger to pilot.

- **Repair Rules**. Once we have identified and explained an expectation failure, we must try to alter our plan to fit the new situation. Again, we have rules for what kinds of changes are permissible. For the air shuttle, we recognize that paying for the ticket on the plane is an acceptable change.

2.5 Psychological Issues

The process model depicted in figure 1 is not meant to stipulate the necessary and sufficient conditions for simulating cognitive behavior. Rather, it is presented as illustrative of a variety of salient issues in case-based reasoning.

We can summarize the psychological assumptions of the case-based reasoning paradigm as follows.

- Memory is predominantly episodic. The primary content of memory is experience.
- Memory is richly indexed. Experiences are related to each other in many complex and abstract ways.
- Memory is dynamic. The organization and structure of memory changes over time.
- Experience guides reasoning. We interpret and understand new situations in terms of prior experience.
- Learning is triggered by failure. When an expectation from a previous case fails to predict a new situation, we learn through incorporating the new episode into memory.

Similarly, we can present the research questions that arise from these respective assumptions.

- What comprises a case? What is the content and structure of an episode in memory? What is the relationship between episodic memory and other types of knowledge? How can we represent Case Memory?

- How is memory organized? What set of indices are appropriate for classifying cases? What search algorithms complement the structure of memory? What are the Indexing Rules?

- How does memory change? What leads to forgetting? How does the memory of cases and stories degrade? How do the Case Memory and Indexing Rules change over time?

- How can we adapt old solutions to new problems? How can we recognize a new situation as being similar to a previous episode? What are the Similarity Metrics and Modification Rules?

- What leads us to reject or accept a new case that is in conflict with a previous case? How do we explain the differences between episodes? How can we learn from mistakes? What are the Repair Rules?

It may seem that we are presenting more questions than answers. However, our basic premise is that case-based reasoning provides a foundation for a broad range of research. It is appropriate and indeed desirable to stimulate research through the principled identification and examination of cognitive phenomena.

We now turn to review the history of case-based reasoning in artificial intelligence research.

3 Computer Models

The first computer programs to use scripts were SAM (Script Applier Mechanism) [Cul78], and FRUMP (Fast Reading Understanding Memory Program) [DeJ79]. These programs read newspaper stories and performed various language tasks, such as translation, summarization, and answering questions. These programs contained static knowledge structures that were used in processing stories. The content of the programs' memory did not change as a result of processing – in spite of the "memory" in FRUMP's name.

These programs were a successful demonstration of natural language processing of stories, and of scripts as a knowledge structure. Understanding a story entailed processing an episode or event. Scripts provided a feasible means for representing such episodic knowledge. However, the programs failed to demonstrate knowledge acquisition. The scripts of SAM and FRUMP were innate, as it were, having been written by programmers. The programs used the scripts to guide processing of stories, but the programs did not learn their scripts through experience.

Furthermore, the programs did not remember anything. SAM or FRUMP could read the same story twenty times in a row and not recognize that it had previously seen that story. Clearly a program that modeled human memory should remember its own experience.

Two programs followed that addressed the issue of memory organization for episodic knowledge: CYRUS and IPP. The first was Kolodner's CYRUS [Kol80, SK79. Kol84, KC86]. The CYRUS program simulated an episodic memory of events relating to former Secretary of State Cyrus Vance. The program would answer questions about a range of "autobiographical" episodes, such as meetings, diplomatic trips, and state dinners. CYRUS was the first program to model episodic storage and retrieval strategies. While the focus of CYRUS was on memory organization and indexing, there was also an attempt to integrate CYRUS with the FRUMP newswire program to provide an automatic update for CYRUS's memory [SKD80]. The combined system, CyFr, would read news stories about the Secretary of State and integrate the events into CYRUS's episodic memory.

Lebowitz' IPP [Leb80] provided a prototype for case-based reasoning and learning programs. IPP read news stories about terrorist acts, such as bombings, kidnappings, and shootings. The program started with generic knowledge about such acts, and after reading hundreds of stories, developed its own set of generalizations about terrorism that it could apply to understanding new stories. We have included a sample protocol from IPP in Appendix A. In that example, the program reads two stories about IRA terrorism in Northern Ireland. The program notices that the victims are establishment, authority figures (policemen and soldiers), and that the terrorists are members of the IRA. IPP then reads a third story about a shooting in Northern Ireland, and the program infers that the unidentified gunman is a member of the IRA.

3.1 Expert Systems: Rules vs. Cases

The programs from the late 1970's that modeled episodic memory were largely natural language processing programs. At that time, another topic of AI research had developed into a primary area of applications, namely rule-based expert systems. Early programs such as DENDRAL [BSF69] and MYCIN [Sho76] had demonstrated the possibility of simulating the problem solving ability of human experts, such as chemists or physicians. The success of these and other programs stimulated interest in developing expert systems for a vast number of technical applications.

The basic unit of knowledge in these expert systems was the *rule.* A rule comprised a conditional test-action pair, e.g., *IF condition, THEN action.* Several hundred rules might be required for handling a typical diagnostic or repair task.

Building rule-based or production systems became a popular enterprise. As experience with expert systems increased, so did awareness of some basic short-comings of the rule-based system paradigm.

The first problem was *knowledge acquisition.* In order to build an expert system, a computer programmer (or knowledge engineer) had to sit down with the human expert informant to determine what rules were appropriate for the given domain. This knowledge was difficult to uncover. The human expert could not simply make a list of the hundreds of rules he used to solve problems. Often the informant would articulate a set of rules that in fact would not accurately reflect his own problem solving behavior. For these reasons, this difficult knowledge acquisition process became known as a bottleneck in constructing rule-based expert systems [HWL83].

Second, the rule-based systems did not have a *memory.* That is, just as SAM and FRUMP would not remember news stories that they had already read, rule-based systems would not remember problems that they had already solved. For example, if a medical diagnosis program is presented with a patient with a certain set of symptoms, the program may fire dozens or hundred or thousands of rules and come up with a diagnosis or treatment. Subsequently, if the program is presented with another patient displaying the same set of symptoms, the program will fire the same set of rules as before. The program will not *remember* having previously seen a similar patient. One might argue that this observation is of little consequence beyond some argument for computational efficiency. However, a program without a memory will not remember its mistakes, and thus, will be destined to repeat them.

Third, rule-based systems were not *robust.* If a problem were presented to the system that did not match any of the rules, the program could not respond. The system's knowledge base was limited to its rules, so if none of the rules could apply, the system had no alternatives.

We may compare the behavior of the rule-based expert system with the behavior of the human expert. The central feature of *expertise* is *experience.* An expert is someone who has vast, specialized experience, who has witnessed numerous cases in the domain, and who has generalized this experience to apply it to new situations. When confronted with a problem, the expert is reminded of previous, similar problems and their respective resolutions. It may be that the expert has so many exemplars for a given problem that the experiences have been distilled into a general rule to be applied.

Thus, the human expert derives knowledge from experience. The basic unit of knowledge is not the *rule*, but the *case*. Human experts acquire knowledge by assimilating new cases, either first-hand or through the reports of others. Furthermore, it is easier for people to articulate knowledge in the form of experience than as rules. This observation suggests the psychological hypothesis that expert knowledge may in fact be encoded primarily as episodes, rather than as rules. We contrast this acquisition of knowledge from experience with the knowledge acquisition bottleneck given above as the first problem of rule-based systems.

Second, human experts remember their own experience. The doctor who fails to treat a case effectively should remember that case when another patient presents the same symptoms. The doctor can learn from his mistakes.

Third, human experts can reason by analogy. If our doctor sees a patient who presents symptoms that are unlike anything in his experience, the doctor need not simply give up. The doctor might be reminded of various previous cases that were similar in one way or another, and devise a treatment accordingly. Just as our first air shuttle trip might remind of us of both an airplane trip and a train ride, the doctor might be able to arrive at a composite diagnosis based on different earlier cases.

These arguments suggest an alternative to the rule-based system: a case-based system. An expert system that can extract information from its experience will be able to grow and acquire knowledge on its own. This is a crucial step for the long-range success of the expert system concept in AI. There are so many tasks to which automated reasoning power might be applied, that it is necessary to develop a mechanism that can assimilate new knowledge directly from experience.

The technology of case-based systems directly addresses problems found in rule-based systems.

- *Knowledge acquisition.* The unit of knowledge is the case, not the rule. It is easier articulate, examine, and evaluate cases than rules.

- *Performance experience.* A case-based system can remember its own performance and modify its future behavior to avoid repeating prior mistakes.

- *Adaptive solutions.* By reasoning from analogy with past cases, a case-based system should be able to construct solutions to novel problems.

The scientific research issues given for case-based reasoning models in section 2.5 also apply directly to the technological research issues for case-based systems. We must answer those same questions in building case-based systems.

- What comprises a case? How can we represent Case Memory?
- How is memory organized? What are the Indexing Rules?
- How does memory change? How do the Case Memory and Indexing Rules change over time?
- How can we adapt old solutions to new problems? What are the Similarity Metrics and Modification Rules?
- How can we learn from mistakes? What are the Repair Rules?

The technology of case-based systems is an instantiation of the psychological theories of case-based reasoning. CYRUS and IPP can be viewed as prototypes for case-based systems. In the 1980's, researchers began explicitly to develop case-based systems.

3.2 Case-based Systems

One of the first case-based expert systems was Simpson's MEDIATOR [KSS85, Sim85]. The program acted as an advisory system for dispute mediation. MEDIATOR would be presented with a dispute situation between two parties and, based on its experiential knowledge base, suggest a resolution. The program addressed problems of similarity measures, memory structures for representing and retrieving cases, adaptation, and recovery from failure.

Simpson proposed the following problem solving principles that characterize case-based reasoning systems [Sim85].

1. Including a capability for case-based reasoning in a problem solving system allows previous computations to be used to suggest solutions to new problems, potentially cutting down the work required to solve a difficult problem from scratch.

2. Case-based reasoning requires access to a dynamic memory capable of storing and retrieving previous experience.

3. Case-based reasoning requires that a problem solver be able to recognize similarity between cases so that only those potentially applicable to the current problem are recalled.

4. Case-based reasoning requires choosing the most appropriate case from a set of potentially applicable ones.

5. Case-based reasoning requires that the problem solver be able to transfer the appropriate information from one case to another.

6. Case-based reasoning requires that the problem solver must receive feedback and be able to evaluate its decisions.

7. Problem solvers must be able to recover from reasoning failures.

Bain applied the case-based approach to legal reasoning. Bain observed lawyers and judges in the context of sentencing convicted criminals. His program, JUDGE [Bai84, Bai86], simulated the process of a judge deciding the appropriate sentence to mete out, based on the judge's experience.

The following is a brief example from JUDGE. The program is given a new case, CRIME8, which has features that remind JUDGE of a previous assault case, CRIME1. JUDGE compares the two cases to decide what sentence should be imposed in the new case. The program explains its reasoning as it goes along.

```
CRIME8 Facts:

First, Terry struck Sam with his fists several times.
        Sam was only slightly hurt.
[Interpreted as UNPROVOKED-VIOLATION]

Next, Sam struck Terry with his fists several times.
        Terry was only slightly hurt.
[Interpreted as PARITY-SELF-DEFENSE with an ACHIEVED-RESULT]

Finally, Terry struck Sam with his fists several times.
        Sam received a concussion.
[Interpreted as PARITY-RETALIATION with a VIOLATIVE-ACCIDENTAL-RESULT.]

THE ONLY THEMATICALLY SIMILAR PREVIOUS SITUATION IN MEMORY WAS
```

```
CRIME1.  CRIME1 CONTAINED A SIMILAR FINAL INTENTION AND RESULT
INTERPRETATION AS CRIME8.

...Comparing CRIME1 from memory with the current input, CRIME8...

The old crime, CRIME1, once again, involved the following:

First, Randy struck Chuck with his fists several times.
        Chuck was only slightly hurt.
[Interpreted as UNPROVOKED-VIOLATION]

Next, Chuck struck Randy with his fists several times.
        Randy was only slightly hurt.
[Interpreted as PARITY-SELF-DEFENSE with an ACHIEVED-RESULT]

Then, Randy slashed at Chuck with a knife one time.
        Chuck's skin was slightly cut.
[Interpreted as ESCALATED-RETALIATION with a BUNGLED-RESULT.]

Next, Chuck slashed at Randy with a knife one time.
        Randy's skin was slightly cut.
[Interpreted as PARITY-SELF-DEFENSE with a BUNGLED-RESULT.]

Finally, Randy stabbed Chuck with a knife several times.
        Chuck died.
[Interpreted as ESCALATED-RETALIATION with an ACHIEVED-RESULT]

THE RESULT WHERE CHUCK WAS KILLED WAS WORSE THAN THE ONE IN WHICH
SAM RECEIVED A CONCUSSION.  NOT ONLY WAS THE OUTCOME IN THE NEWER
SITUATION ACCIDENTAL, BUT RANDY BOTH INTENDED AND CAUSED MORE HARM
THAN TERRY.

THE INTENT OF RANDY, THE OFFENDER IN THE OLDER CRIME, TO ACT
REPEATEDLY TO STAB CHUCK TO DEATH WAS MORE SEVERE THAN THAT OF
TERRY, THE OTHER OFFENDER, TO ACT REPEATEDLY TO HIT SAM.  IN
ADDITION, ALTHOUGH NEITHER ACTOR'S INTENTIONS WERE JUSTIFIED,
RANDY ESCALATED THE LEVEL OF VIOLENCE, WHILE TERRY DID NOT.

COMPARISON FINISHED WITH RESULT THAT THE OLD CRIME,
CRIME1, WAS WORSE.

THE SENTENCE ORIGINALLY GIVEN FOR CRIME1, FOR VIOLATION OF MURDER,
WAS A TERM OF IMPRISONMENT OF NOT LESS THAN 40 YEARS NOR MORE THAN
50 YEARS TO BE SERVED IN THE STATE'S PRISON.  SINCE THE
```

Program	Year	Domain	Task
CYRUS	1980	Diplomacy	Memory organization and retrieval
IPP	1980	Terrorism	Story understanding / learning
MEDIATOR	1985	Mediation	Case-based problem solving
JUDGE	1986	Law	Case-based assessment
CHEF	1986	Cooking	Case-based planning / learning

Table 1: Early case-based systems

```
CIRCUMSTANCES OF CRIME1 ARE FOUND TO BE WORSE THAN CRIME8, DUE TO
MORE SEVERE ACTIONS AND RESULTS AT THE END OF THE CRIME, THE
OFFENDER WILL GET A SENTENCE WHICH IS MORE LENIENT THAN THE
SENTENCE FOR CRIME1, RELATIVE TO THE CRIME OF ASSAULT IN THE
SECOND DEGREE.

THE SENTENCE TO BE GIVEN FOR CRIME8, FOR VIOLATION OF ASSAULT IN
THE SECOND DEGREE AS DEFINED IN SECTION 53A-60 IN THE CONNECTICUT
PENAL CODE, WILL BE A TERM OF IMPRISONMENT OF NOT LESS THAN 3
YEARS TO BE SERVED IN THE STATE'S PRISON.  THE MAXIMUM PERIOD OF
INCARCERATION ALLOWED FOR VIOLATION OF THIS STATUTE IS 5 YEARS.
```

This example illustrates some of the main aspects of case-based reasoning. First, JUDGE has a case library comprising an episodic memory of previously adjudicated crimes. Second, new cases are analyzed to provide indices for retrieving past cases. Third, JUDGE can compare the new case with the old case to determine how the new situation may require modification of the prior sentence. Fourth, JUDGE provides an explanation of its reasoning by explicit analysis and comparison of the two cases. Finally, the new case becomes assimilated into JUDGE's case library for future use.

Hammond's CHEF program [Ham84, Ham86] developed new plans based on its own experience in the domain of cooking. When faced with the task of preparing a dish for which it had no appropriate plan (recipe), CHEF would modify an existing plan to fit the new situation, and then try to detect and correct any errors that resulted. CHEF would learn from its own mistakes. An annotated example from CHEF is shown in Appendix B.

Table 1 summarizes these early programs, all of which represent completed Ph.D. research. Recent work in case-based reasoning has proceeded apace, reflecting a wide-spread and growing interest in the case-based paradigm. Current case-based research projects are summarized in table 2. This list is not exhaustive, merely illustrative. These efforts demonstrate a wide range

Program	Citation	Site	Domain	Task
PLEXUS	[Alt86]	Berkeley	Travel	Adaptive planning
CBD	[HH88]	Chicago	Machines	Case-based diagnosis
TRUCKER	[MHC88]	Chicago	Scheduling	Pluralistic planning
CYCLOPS	[Nav88]	CMU	Landscaping	Design problem solving
PERSUADER	[Syc88]	CMU	Mediation	Plan adaptation and repair
PRODIGY	[CV88]	CMU	Algebra	Derivational analogy
CBR Shell	[Rie88]	CSI	Tool	Programming shell
JULIA	[Shi88]	GA Tech	Cooking	Analogical reasoning
MEDIC	[Tur88]	GA Tech	Medicine	Diagnostic reasoning
PARADYME	[Kol88]	GA Tech	Cooking	Parallel memory retrieval
n/a	[BM88]	Lockheed	Machines	Explanation-based learning
CASEY	[Kot88]	MIT	Medicine	Reasoning about evidence
n/a	[WDH88]	TI	Military	Tactical planning
JOHNNY	[Sta88]	TMC	Reading	Memory-based reasoning
CBS	[BL88]	UMass	Puzzles	Case-based search
HYPO	[RA88]	UMass	Law	Case-based reasoning
TA	[Wil88]	UMass	Programming	Case-based learning
ANON	[Owe88]	Yale	Proverbs	Indexing prototypical cases
DECIDER	[Far88]	Yale	History	Case-based teaching
DMAP	[RM85]	Yale	Economics	Direct memory access parsing
IVY	[Hun85]	Yale	Medicine	Case-based diagnosis
SWALE	[KL88]	Yale	Post-mortem	Case-based explanations

Table 2: Current case-based research projects

of domains and research issues that have flowed from the case-based reasoning paradigm.

4 Summary

In this paper we have reviewed some of the beginnings, motivations, and trends in case-based reasoning research. Case-based reasoning grew out of psychological models of episodic memory and the technological impetus of artificial intelligence. In recent years, interest in case-based reasoning has grown across the country.

The two long-term agendas of case-based reasoning remain: to develop a scientific model of human memory, and to build robust computer programs that can assimilate experiences and adapt to new situations. As the results of the past few years seem to demonstrate, these enterprises appear to be synergistic.

5 References

[Alt86] R. Alterman. An adaptive planner. In *Proceedings of the Fifth National Conference on Artificial Intelligence*, pages 65–69, AAAI, Philadelphia, PA, August 1986.

[Bai84] W.M. Bain. *Toward a Model of Subjective Interpretation.* Technical Report 324, Yale University Department of Computer Science, July 1984.

[Bai86] W.M. Bain. *Case-based Reasoning: A Computer Model of Subjective Assessment.* PhD thesis, Yale University, 1986. Technical Report 470.

[Bar32] F.C. Bartlett. *Remembering.* University Press, Cambridge, 1932.

[BBT79] G.H. Bower, J.B. Black, and T.J. Turner. Scripts in memory for text. *Cognitive Psychology*, 11:177–220, 1979.

[BL88] S. Bradtke and W. Lehnert. Some experiments with case-based search. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 80–93, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[BM88] R. Barletta and W. Mark. Explanation-based indexing of cases. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 50–60, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[BSF69] B.G. Buchanan, G.L. Sutherland, and E.A. Feigenbaum. Heuristic dendral: a program for generating explanatory hypotheses in organic chemistry. In *Machine Intelligence 4*, Edinburgh University Press, 1969.

[CQ69] A.M. Collins and M.R. Quillian. Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behavior*, 8:240–247, 1969.

[Cul78] R. Cullingford. *Script Application: Computer Understanding of Newspaper Stories.* PhD thesis, Yale University, 1978. Technical Report 116.

[CV88] J. Carbonell and M. Veloso. Integrating derivational analogy into a general problem solving architecture. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 104–124, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[DeJ79] G. DeJong. *Skimming Stories in Real Time: An Experiment in Integrated Understanding.* PhD thesis, Yale University, May 1979. Technical Report 158.

[Far88] R. Farrell. Facilitating self-education by questioning assumptive reasoning using paradigm cases. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 136–153, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Ham84] K. Hammond. *Indexing and Causality: The organization of plans and strategies in memory.* Technical Report 351, Yale University Department of Computer Science, December 1984.

[Ham86] K.J. Hammond. *Case-based Planning: An Integrated Theory of Planning, Learning and Memory.* PhD thesis, Yale University, 1986. Technical Report 488.

[HH88] K.J. Hammond and N. Hurwitz. Extracting diagnostic features from explanations. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 169–178, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Hun85] L. Hunter. Steps toward building a dynamic memory. In *Proceedings of the Third International Machine Learning Workshop*, pages 70–73, Office of Naval Research, Skytop, PA, June 1985.

[HWL83] F. Hayes-Roth, D.A. Waterman, and D.B. Lenat (eds.). *Building Expert Systems.* Addison-Wesley, Reading, Mass., 1983.

[KC86] J.L. Kolodner and R. Cullingford. Towards a memory architecture that supports reminding. In *Proceedings of the Eighth Annual Conference of the Cognitive Science Society*, Cognitive Science Society, Amherst, MA, August 1986.

[Kin72] W. Kintsch. Notes on the structure of semantic memory. In *Organization of Memory*, Academic Press, New York, 1972.

[KL88] A.M. Kass and D.B. Leake. Case-based reasoning applied to constructing explanations. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 190–208, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Kol80] J.L. Kolodner. *Retrieval and Organizational Strategies in Conceptual Mcmory: A Computer Model.* PhD thesis, Yale University, November 1980. Technical Report 187.

[Kol84] J.L. Kolodner. *Retrieval and Organizational Strategies in Conceptual Memory.* Lawrence Erlbaum Associates, Hillsdale, N.J., 1984.

[Kol88] J.L. Kolodner. Retrieving events from a case memory: a parallel implementation. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 233–249, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Kot88] P. Koton. Reasoning about evidence in causal explanations. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 260–270, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[KSS85] J.L. Kolodner, R.L. Simpson., and K. Sycra-Cyranski. A process model of case-based reasoning in problem-solving. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, IJCAI, Los Angeles, CA., August 1985.

[Leb80] M. Lebowitz. *Generalization and Memory in an Integrated Understanding System.* PhD thesis, Yale University, October 1980. Technical Report 186.

[MHC88] M. Marks, K.J. Hammond, and T. Converse. Planning in an open world: a pluralistic approach. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 271–285, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Min75] M. Minsky. A framework for representing knowledge. In P. Winston, editor. *The Psychology of Computer Vision*, chapter 6, pages 211–277, McGraw-Hill, New York, 1975.

[Nav88] D. Navinchandra. Case-based reasoning in cyclops, a design problem solver. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 286–301, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Owe88] C. Owens. Domain-independent prototype cases for planning. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 302–311. Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Qui68] M.R. Quillian. Semantic memory. In M. Minsky, editor, *Semantic Information Processing*, MIT Press, Cambridge, MA., 1968.

[RA88] E.L. Rissland and K.D. Ashley. Credit assignment and the problem of competing factors in case-base reasoning. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 327–344, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[RB87] C.K. Riesbeck and W.M. Bain. A methodology for implementing case-based reasoning systems. 1987. A contracted report submitted to Lockhead from Cognitive Systems.

[Rie88] C.K. Riesbeck. An interface for case-based knowledge acquisition. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 312–326, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[RLN72] D.E. Rumelhart, P.H. Lindsay, and D.A. Norman. A process model for long-term memory. In *Organization of Memory*, Academic Press, New York, 1972.

[RM85] C.K. Riesbeck and C.E. Martin. *Direct Memory Access Parsing.* Technical Report 354, Yale University Department of Computer Science, January 1985.

[SA75] R.C. Schank and R. Abelson. Scripts, plans, and knowledge. In *Proceedings of the Fourth International Joint Conference on Artificial Intelligence*, IJCAI, Tbilisi, USSR, 1975.

[SA77] R.C. Schank and R. Abelson. *Scripts, Plans, Goals and Understanding.* Lawrence Erlbaum Associates, Hillsdale, New Jersey, 1977.

[Sch72] R.C. Schank. Conceptual dependency: a theory of natural language understanding. *Cognitive Psychology*, 3(4):552–631, 1972.

[Sch75a] R.C. Schank. *Conceptual Information Processing.* Volume 3 of *Fundamental Studies in Computer Science*, North-Holland, Amsterdam, 1975.

[Sch75b] R.C. Schank. The structure of episodes in memory. In *Representation and Understanding*, pages 237–272, Academic Press, New York, 1975.

[Sch79] R.C. Schank. *Reminding and Memory Organization: An Introduction to MOPs.* Technical Report 170, Yale University Department of Computer Science, 1979.

[Sch80] R.C. Schank. Language and memory. *Cognitive Science*, 4(3):243–284, 1980.

[Sch81] R.C. Schank. Failure-driven memory. *Cognition and Brain Theory*, 4(1):41–60, 1981.

[Sch82] R.C. Schank. *Dynamic memory: A theory of learning in computers and people.* Cambridge University Press, 1982.

[Sch86a] R.C. Schank. *Explanation Patterns: Understanding Mechanically and Creatively.* Lawrence Erlbaum Associates, Hillsdale, NJ, 1986.

[SCH86b] R.C. Schank, G. Collins, and L. Hunter. Transcending inductive category formation in learning. *Behavioral and Brain Sciences*, 9(4), 1986.

[Shi88] H.S. Shinn. Abstractional analogy: a model of analogical reasoning. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 370–387. Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Sho76] E.H. Shortliffe. *Computer-based medical consultations: MYCIN.* American Elsevier, New York, 1976.

[Sim85] R.L. Simpson. *A Computer Model of Case-based Reasoning in Problem-solving: An Investigation in the Domain of Dispute Mediation.* PhD thesis, School of Information and Computer Science, Georgia Institute of Technology, 1985.

[SK79] R. Schank and J.L. Kolodner. *Retrieving information from and episodic memory, or Why computers memories should be more like people's.* Technical Report 159, Yale University Department of Computer Science, 1979.

[SKD80] R.C. Schank, J.L. Kolodner, and G.F. DeJong. *Conceptual Information Retrieval.* Technical Report 190, Yale University Department of Computer Science, 1980.

[Sta88] C. Stanfill. Learning to read: a memory-based model. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 402–413, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Syc88] K. Sycara. Using case-based reasoning for plan adaptation and repair. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 425–434, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Tul72] E. Tulving. Episodic and semantic memory. In *Organization of memory*, chapter 10, pages 381–403, Academic Press, Inc., New York, 1972.

[Tul83] E. Tulving. *Elements of Episodic Memory.* Oxford University Press, Oxford, 1983.

[Tur88] R.M. Turner. Organizing and using schematic knowledge for medical diagnosis. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 435–446, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[WDH88] R.S. Wall, D. Donahue, and S. Hill. The use of domain semantics for retrieval and explanation in case-based reasoning. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 447–462, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Wil88] R.S. Williams. Learning to program by examining and modifying cases. In *Proceedings of DARPA Workshop on Case-Based Reasoning*, pages 463–474, Defense Advanced Research Projects Agency, Information Science and Technology Office, Clearwater, FL, May 1988.

[Woo75] W.A. Woods. What's in a link: foundations for semantic networks. In *Representation and Understanding*, chapter 2, pages 35–82, Academic Press, New York, 1975.

A Appendix: Case-based learning in IPP

The program IPP [Leb80] demonstrates the role of reminding in learning. IPP was a natural language program. Its acronym stood for *Integrated Partial Parsing*, which relates to how IPP would convert English sentences into a conceptual representation.

Below we present a transcript of IPP reading three news stories about shootings in Northern Ireland.

```
IPP created 13-Feb-81 13:11:21, ready 25-Jul-85 14:10:21

*(PARSE XX1)

Story: XX1 (4 12 79) NORTHERN-IRELAND NONE

(IRISH REPUBLICAN ARMY GUERRILLAS AMBUSHED A MILITAPY PATROL IN
WEST BELFAST YESTERDAY KILLING ONE BRITISH SOLDIER AND BADLY
WOUNDING ANOTHER ARMY HEADQUARTERS REPORTED)

Processing:

IRISH REPUBLICAN ARMY
              : Phrase
IRA           : Token refiner - save and skip
GUERRILLAS    : Interesting token - GUERRILLAS OF THE IRA


Instantiated I-TERRORISM structure

Predictions - I-TERRORISM-SYN-FINDER I-TERRORISM-SUB-STRUCTURE
              INFER-I-TERRORISM-SUB-STRUCTURES
              REDUNDANT-SCRIPT-WORDS DEFAULT-ORGANIZATION
              DEFAULT-VICTIM-TYPE COUNTER-MEASURES

>>> Beginning memory update ...

New features:  EVO (XX1) (I-TERRORISM)
  ACTOR           ORG          IRA
                  POL-POS      BAD-GUY
                  AFFILIATION  CATHOLIC
```

```
 LOCATION        AREA         WESTERN-EUROPE
                 NATION       *NORTHERN-IRELAND*

Best existing S-MOP(s) --
I-TERRORISM
```

At this point, IPP has started reading story XX1. IPP recognizes that it is an instance of terrorism. There are a number of things that IPP expects to see in the rest of the story dealing with typical terrorist activities. The program begins to build a conceptual representation of the story, labeled EV0, which at this point includes just the actor and location of the event.

The program proceeds in a similar fashion to the end of the story, producing the following conceptual representation.

```
Indexing  EV0 (XX1) as variant of I-TERRORISM

>>> Memory incorporation complete

Story Representation:

** MAIN EVENT **
EV0 =
 MEM-NAME    I-TERRORISM
 ACTOR       GUERRILLAS OF THE IRA
 VICTIM      IRISH MILITARY PATROL
 INSTANCES
    EV2 =
     MEM-NAME    S-ATTACK-PERSON
     ACTOR       GUERRILLAS OF THE IRA
     VICTIM      IRISH MILITARY PATROL
     METHODS
        EV1 =
         MEM-NAME    $AMBUSH
         ACTOR       GUERRILLAS OF THE IRA
         VICTIM      IRISH MILITARY PATROL
     RESULTS
        EV3 =
         MEM-NAME    CAUSE-DEATH
         ACTOR       GUERRILLAS OF THE IRA
         VICTIM      IRISH MILITARY PATROL
         HEALTH      -10
        EV4 =
         MEM-NAME    CAUSE-WOUND
         ACTOR       GUERRILLAS OF THE IRA
         HEALTH      -5
 TIME        YESTERDAY

5121 msec CPU (0 msec GC), 41000 msec clock, 7417 conses
NIL
```

IPP's representation of death (HEALTH = -10) was somewhat prosaic. However, the program basically got the facts right and, more importantly, would remember what it had read.

Next, we give IPP a second story, XX2, about Northern Ireland.

```
*(PARSE XX2)

Story: XX2 (11 11 79) NORTHERN-IRELAND NONE

(A SUSPECTED IRISH REPUBLICAN ARMY GUNMAN KILLED A 50 -YEAR-OLD
UNARMED SECURITY GUARD IN EAST BELFAST EARLY TODAY THE POLICE SAID)

Processing:

A              : Function word - Token refiner - save and skip
SUSPECTED      : Dull verb - skipped
IRISH REPUBLICAN ARMY
               : Phrase
IRA            : Token refiner - save and skip
GUNMAN         : Interesting token - GUNMAN OF THE IRA

Predictions - TERRORISM ROBBERY KIDNAPPING-SCRIPT
              LOOK-FOR-GUNMAN-ASSOCIATED-SCRIPT
              FIND-GUNMAN-ASSOC-SIBLING

KILLED         : Word satisfies prediction

Prediction confirmed - FIND-GUNMAN-ASSOC-SIBLING

Instantiated CAUSE-DEATH structure
```

```
Predictions - REDUNDANT-SCRIPT-WORDS INFER-REASON-FOR-DEATH
              FIND-REASON-FOR-DEATH IMPORTANT-VICTIM
              CAUSE-DEATH-ROLE-FINDER END-ROLE-FINDER
```

IPP continues in this vein, but along the way, the program is reminded of the first story (XX1). It then creates a Memory Organization Packet (MOP [Sch82]) noting the similarities between the two stories.

```
Creating more specific S-ATTACK-PERSON (SpM3) from events
EV2 (XX1) EV6 (XX2) with features:

VICTIM     (1)    ROLE           AUTHORITY
                  POL-POS        ESTAB
ACTOR      (1)    ORG            IRA
                  POL-POS        BAD-GUY
                  AFFILIATION    CATHOLIC
RESULTS    (1)    AU             HURT-PERSON
                  HEALTH         -10
LOCATION   (1)    AREA           WESTERN-EUROPE
                  NATION         *NORTHERN-IRELAND*
I-MOP      (1)    I-MOP          I-TERRORISM

Reminded of:  EV2 (XX1) (MOP creation)

>>> Memory incorporation complete

Story Representation:

** MAIN EVENT **
EV8 =
 MEM-NAME    I-TERRORISM
 ACTOR       GUNMAN OF THE IRA
 VICTIM      50 YEAR OLD IRISH UNARMED GUARD
 PLACE       EAST BELFAST
 INSTANCES
    EV6 =
     MEM-NAME    S-ATTACK-PERSON
     ACTOR       GUNMAN OF THE IRA
     VICTIM      50 YEAR OLD IRISH UNARMED GUARD
     RESULTS
        EV5 =
         MEM-NAME    CAUSE-DEATH
         ACTOR       GUNMAN OF THE IRA
         VICTIM      50 YEAR OLD IRISH UNARMED GUARD
         HEALTH      -10
         PLACE       EAST BELFAST
     METHODS
        EV7 =
         MEM-NAME    $SHOOT
         ACTOR       GUNMAN OF THE IRA
         VICTIM      50 YEAR OLD IRISH UNARMED GUARD
         PLACE       EAST BELFAST
     PLACE       EAST BELFAST
 TIME        TODAY

3464 msec CPU (0 msec GC), 14000 msec clock, 6677 conses
NIL
```

Finally, we give IPP a third story about a shooting in Northern Ireland, but in this case the gunman is not identified.

```
*(PARSE XX3)

Story: XX3 (1 12 80) NORTHERN-IRELAND NONE

(A GUNMAN SHOT AND KILLED A PART-TIME POLICEMAN AT A SOCCER
MATCH SATURDAY AND ESCAPED THROUGH THE CROWD TO A WAITING
GETAWAY CAR *COMMA* POLICE SAID)

Processing:

A             : Function word - Token refiner - save and skip
GUNMAN        : Interesting token - GUNMAN

Predictions - TERRORISM ROBBERY KIDNAPPING-SCRIPT
              LOOK-FOR-GUNMAN-ASSOCIATED-SCRIPT
              FIND-GUNMAN-ASSOC-SIBLING

SHOT          : Word satisfies prediction
```

```
Prediction confirmed - LOOK-FOR-GUNMAN-ASSOCIATED-SCRIPT
```

IPP reads along some more, but when it starts to update memory, it comes across the Specification MOP that it had previously created from the first two stories.

```
>>> Beginning memory update ...

New features:  EV10 (XX3) (S-ATTACK-PERSON)
  ACTOR           POL-POS       BAD-GUY
  METHODS         AU            $SHOOT
  LOCATION        AREA          WESTERN-EUROPE
                  NATION        *NORTHERN-IRELAND*

Best existing S-MOP(s) --
SpM3 -- potential remindings: EV6 (XX2)

Predicted features (SpM3)
  RESULTS         HEALTH        -10
                  AU            HURT-PERSON
  ACTOR           AFFILIATION   CATHOLIC
                  ORG           IRA
  VICTIM          POL-POS       ESTAB
                  ROLE          AUTHORITY
  I-MOP           I-MOP         I-TERRORISM

>>> Memory update complete
```

IPP continues, and when it finishes the story, IPP incorporates the features from its previous generalization. Specifically, IPP infers that the gunman is a member of the IRA.

```
Indexing  EV10 (XX3) as variant of SpM3

Reminded of  EV6 (XX2) (Last MOP reference)

Reminded of  EV6 (XX2) (varies from MOP in same way)

Adding default feature ACTOR ORG IRA to EV10
```

```
>>> Memory incorporation complete

Story Representation:

** MAIN EVENT **
EV10 =
 MEM-NAME    S-ATTACK-PERSON
 ACTOR       GUNMAN OF THE IRA
 VICTIM      PART-TIME POLICEMAN AT SOCCER MATCH
 METHODS
    EV9 =
     MEM-NAME    $SHOOT
     ACTOR       GUNMAN OF THE IRA
     VICTIM      PART-TIME POLICEMAN AT SOCCER MATCH
 RESULTS
    EV11 =
     MEM-NAME    CAUSE-DEATH
     ACTOR       GUNMAN OF THE IRA
     VICTIM      PART-TIME POLICEMAN AT SOCCER MATCH
     HEALTH      -10
 TIME        SATURDAY
 SCENES
    EV12 =
     MEM-NAME    SS-ESCAPE
     ACTOR       GUNMAN OF THE IRA

5926 msec CPU (1887 msec GC), 27000 msec clock, 7252 conses
NIL
```

The timings at the end of each run indicate the speed of IPP. Note that most each of the stories were processed in under a minute, with only a few seconds of actual computer time. The program was written in UCI-LISP and ran on a DEC 2060 time-shared mainframe.

B Appendix: Case-based planning in CHEF

CHEF [Ham84, Ham86] is a computer program in the domain of cooking which generates original plans, which take the form of recipes, by modifying existing plans. It demonstrates how episodic knowledge can be used to guide planning and avoid past failures.

When presented with a problem – how to prepare a certain dish – the program is reminded of previous related recipes. It modifies the most similar previous recipe to fit the new requirements, and then tries out the new recipe. CHEF tests the recipe through a simulation involving rules which specify the physical effects of each step of the cooking process. The results are then examined to see if they match the goals of the intended dish. If the program recognizes a failure, it then tries to analyze and explain the failure through a process of reasoning by asking questions. Finally, the program modifies the recipe in light of its explanation to correct the failure. This case-based planning process closely follows the flow chart given in figure 1.

In the following example, the program has been asked by the user to make a souffle with strawberries. We present annotated output from the program.

```
Searching for plan that satisfies -
   Include strawberry in the dish.
   Make a souffle.

Found recipe -> REC4 VANILLA-SOUFFLE

Recipe exactly satisfies goals ->
    Make a souffle.

Recipe must be altered to match ->
    Include strawberry in the dish.

Building new name for copy of VANILLA-SOUFFLE based on its goals.

Calling recipe STRAWBERRY-SOUFFLE
...
```

The program is reminded of a previous related recipe for Vanilla Souffle. It then copies and modifies the recipe to add strawberries. The computer simulates the execution of the recipe to see if its goals are satisfied. It discovers that most of the goals are met: the batter is baked, the dish tastes like berries, the dish is sweet, etc. However, there are some important goals that failed.

```
Checking for negative features ->

Unfortunately: The batter is now a bad texture.
In that: The batter is now flat.

Recipe -> STRAWBERRY-SOUFFLE has failed goals.

The goal: The batter is now risen.
is not satisfied.

Changing name of recipe STRAWBERRY-SOUFFLE

to BAD-STRAWBERRY-SOUFFLE
```

The program now tries to *learn* from this failure by asking questions and explaining why the failure happened.

```
ASKING THE QUESTION: 'What is the failure?'
ANSWER-> The failure is: It is not the case that:
    The batter is now risen.

ASKING THE QUESTION: 'What is the preferred state?'
ANSWER-> The preferred state is: The batter is now risen.

ASKING THE QUESTION: 'What was the plan to achieve the
    preferred state?'
ANSWER-> The plan was: Bake the batter for twenty five minutes.

ASKING THE QUESTION: 'What were the conditions that led
    to the failure?'
ANSWER-> The condition was: There is an imbalance between
    the whipped stuff and the thin liquid.

         ************ noting balance failure ************

Only one aspect of the imbalance:
 There is an imbalance between the whipped stuff and the thin liquid.
is unexpected.

The state:
 There is whipped stuff in the bowl from the total equaling 60 teaspoons.
normally participates in the goal:
 The batter is now risen.

Only the other aspect of the imbalance:
 There is thin liquid in the bowl from the strawberry equaling
    2.4 teaspoons
is an unexpected condition.
```

The program continues its analysis and eventually decides to modify the recipe to reduce the imbalance between the whipped egg whites and the liquid from the strawberry mixture. The program uses Thematic Organization Packets (TOPs [Sch82]) to make the high level association necessary to alter the plan. It has five strategies which it considers.

```
Found TOP TOP3 -> SIDE-EFFECT:DISABLED-CONDITION:BALANCE
TOP -> SIDE-EFFECT:DISABLED-CONDITION:BALANCE has 5
       strategies associated with it:

       USE-DIFFERENT-PLAN:PLAN1
       USE-DIFFERENT-PLAN:PLAN2
       ADJUNCT-PLAN
       RECOVER
       ADJUST-BALANCE

Applying TOP -> SIDE-EFFECT:DISABLED-CONDITION:BALANCE
to failure It is not the case that: The batter is now risen.
in recipe BAD-STRAWBERRY-SOUFFLE

Asking questions needed for evaluating strategy:
    USE-DIFFERENT-PLAN:PLAN1

ASKING ->
Is there an alternative to
 Pulp the strawberry.
that will enable
```

```
 The dish now tastes like berries.
which does not cause
 There is thin liquid in the bowl from the strawberry
    equaling 2.4 teaspoons

Found plan: Instead of doing step: Pulp the strawberry
 do: Using the strawberry preserves.
```

The program continues to evaluate the other four strategies, compares the results, and finally decides to apply the ADJUST-BALANCE strategy, which in this case means to add more egg whites.

```
Changing name of recipe BAD-STRAWBERRY-SOUFFLE
to STRAWBERRY-SOUFFLE

Implementing plan -> Increase the amount of egg white used.
Suggested by strategy ADJUST-BALANCE

New recipe is -> STRAWBERRY-SOUFFLE

               STRAWBERRY-SOUFFLE

          Two teaspoons of vanilla
          A half cup of flour
          A quarter cup of sugar
          A quarter teaspoon of salt
          A half cup of milk
          Two cups of milk
          One piece of vanilla bean
          A quarter cup of butter
          Five egg yolks
          Six egg whites
          One cup of strawberry

Mix the flour with the sugar and salt.
Mix the milk with the mixture of sugar, salt and flour.
Boil the milk and vanilla bean for less than a half minute.
Remove the vanilla bean from the milk.
Mix the mixture of milk, sugar, salt and flour with the milk.
Simmer the mixture of milk, sugar, salt and flour for five minutes.
```

```
Whip the egg yolk.
Add the butter and mixture of egg yolk to the mixture of milk,
 sugar, salt and flour.
Cool the mixture of egg yolk, milk, sugar, salt, flour and butter.
Whip the egg white.
Add the vanilla and mixture of egg white to the mixture of egg yolk,
 milk, sugar, salt, flour and butter.
Pulp the strawberry.
Mix the strawberry with the spices, egg, milk, salt, flour and
 butter.
Pour the mixture of egg, spices, strawberry, salt, milk, flour
 and butter into a nine inch baking-dish.
Bake the batter for twenty five minutes.

If this plan is successful, the following should be true:

         The batter is now baked.
         The batter is now risen.
         The dish now tastes like berries.
         The dish now tastes sweet.
         The dish now tastes like vanilla.
```

Once all new interactions have been validated, the new recipe is indexed in the data base in terms of those interactions. When the program is then given the task of creating a raspberry souffle, it is reminded of the dish it just created and produces a new recipe which has no failures. The program has learned by analyzing failures through asking questions.

The CHEF program illustrates case-based reasoning and learning. The program was presented with a situation for which it had no exact previous match. It was *reminded* of a similar previous case, and used that as the basis for solving the new problem. In the course of adapting that previous case to fit the new situation, the program encountered several additional problems, but was able to respond to those on the basis of previous cases as well. This is an example of creative reasoning. When it encountered failures, the program was able to reason by asking questions. That is, it explained the failures through a question-based reasoning chain. These reasoning mechanisms contribute to the adaptive learning ability and creativity of the program.